NASA Electronic Library System (NELS)

19P

The System Impact of Security

Version 1.0

IN-82-CR
OCIT
191248

Terry L. McGregor

I-NET, Inc.

7/93

Cooperative Agreement NCC 9-16
Research Activity No. RB.08

NASA Johnson Space Center
Information Systems Directorate
Information Technology Division

ricis

Research Institute for Computing and Information Systems
University of Houston-Clear Lake

DELIVERABLE REPORT

RICIS Preface

This research was conducted under auspices of the Research Institute for Computing and Information Systems by Terry L. McGregor of I-NET, Inc. Dr. E. T. Dickerson served as RICIS research coordinator.

Funding was provided by the Information Technology Division, Information Systems Directorate, NASA/JSC through Cooperative Agreement NCC 9-16 between the NASA Johnson Space Center and the University of Houston-Clear Lake. The NASA research coordinator for this activity was Ernest M. Fridge III, Deputy Chief of the Software Technology Branch, Information Technology Division, Information Systems Directorate, NASA/JSC.

The views and conclusions contained in this report are those of the author and should not be interpreted as representative of the official policies, either express or implied, of UHCL, RICIS, NASA or the United States Government.

NASA ELECTRONIC LIBRARY SYSTEM

THE SYSTEM IMPACT OF SECURITY

Version 1.0

July 1993

NASA ELECTRONIC LIBRARY SYSTEM

THE SYSTEM IMPACT OF SECURITY

Prepared by

Terry L. McGregor

I-NET, Inc.

July 1993

1.0 INTRODUCTION

This paper discusses security issues as they relate to the NASA Electronic Library System which is currently in use as the repository system for AdaNET System Version 3 (ASV3) being operated by MountainNET, Inc.

NELS was originally designed to provide for public, development, and secure collections and objects. The secure feature for collections and objects was deferred in the initial system for implementation at a later date. The NELS system is now 9 months old and many lessons have been learned about the use and maintenance of library systems. MountainNET has 9 months of experience in operating the system and gathering feedback from the ASV3 user community. The user community has expressed an interest in seeing security features implemented in the current system. The time has come to take another look at the whole issue of security for the NELS system.

Two requirements involving security has been put forth by MountainNET for the ASV3 system. The security requirements are listed below.

1.) Incorporate at the collection level a security scheme to allow restricted access to collections. This should be invisible to end users and be controlled by librarians.

2.) Allow inclusion of applications which can be executed only by a controlled group of users, for example, an application which can be executed by librarians only.

The requirements provide a broad framework in which to work, These requirements raise more questions than answers,. To explore the impact of these requirements a top down approach will be used. Starting at the application layer and later looking at the impact of the changes to the underlying application structure.

1.1 SECURITY

Security in NELS will be supported by the addition of two (2) new tables in the library repository. The first table is the users table and contains information about the users of the system. The second table is the group table and allows administrators to create group-s of users in the NELS system. The notion of groups in the NELS system will allow administrators to restrict access to specific collections to specific users.

1.1.1 Terminology

The terms used to describe the security features are defined in the following section.

Groups:

A group is a set of one or more users who share the same access privileges in the NELS system. Groups are used to grant or restrict access to collections and applications.

Output Requests:
Output requests are methods for retrieving information from the NELS system. Users may request objects and metadata from the repository be transfered to their local host computer electronically by network (using File Transfer Prototol [FTP]) or phone line (using KERMIT). Users may also request the information be sent to off-line media such as magnetic tape, floppy disk, or printer and sent by mail. Output requests are available to all user os the NELS system.

Public Collections:
A public collection is a logical grouping of objects which are accessible to all users of the NELS system. A public collection is a member of zero(0) or one(1) group (i.e., the public group).

Public User:
A public user is a system user which is a member of zero(0) or one (1) groups (i.e., the public group). A public user's access is restricted to public collections.

Public Resident Application:
A public resident application is a program which resides on the NELS host system and may be called by users from the NELS Applications menu. Public resident applications are available to all user o the NEIS system./

Search Domain
A users search domain is the set of collections and objects which is available to the user for searching, The search domain is specifically defined by:

1.) The user's position in the collection hierarchy.

2.) The user's search strategy (i.e., current collection, sub-collections, or related collections.)

3.) The set of public and secure collections accessible by the user based on their position in the collection hierarchy and search strategy.

Secure Resident Application:
A secure resident application is a program which resides on the NELS host system and may be called by users of the NELS application menu. A secure resident application is a member of one (1) or more groups, all not public, and is only available to authorized users.

Secure Collection:
A secure collection is a logical grouping of objects which are accessible to an authorized subset of the NELS users., A secure collection is a member of one (1) or more not public groups.

Secure User:
A secure user is a system user which is a member of one (1) or more not public groups. a secure user has access to all public collections and authorized secure collections. a secure user may also have access to authorized secure applications.

2.0 THE IMPACT OF SECURITY ON THE NELS USER INTERFACE

This section discusses the impact on the user interface of incorporating security into the NELS library system. The change to each window is explained below.

The typical user will notice little change in the user interface of NELS as a result of implementing security features. the largest impact to the user interface will be seen by the NELS system librarians. Support for security features will result in major changes to existing administrative functionality. This change includes the addition of two (2) NELS administration sub-systems which are explained in the administration section.

2.1 Collection Browser

The incorporation of security at the collection level will be invisible to the user. The user will not be able to distinguish between secure and public collections. The public user of the NELS system will see only public collections. As the public user is browsing the collection hierarchy, no secure collection will be visible in the collection viewer. This will be consistent, independent of the collection browsing mode. This feature will be an integral part of the collection browsing mechanism.

The secure user of the NELS system will have access to all public collections and authorized secure collections. Any secure collections which the user does not have approved access to will not appear in the collection browser. This will be consistent, independent of the browsing mode. This feature will be an integral part of the collection browsing mechanism

2.2 Object Browser

The object browser will list all objects contained in the in the collection(s) accessible by the users based on their search strategy. For the public user the objects listed will be from those public collections in the search domain. For the secure user the objects listed will come from public collections and authorized secure collections in the search domain. From the object browser the user will be able to view abstracts, metadata, and object files., To add additional flexibility to the viewing of objects, viewers will be associated with collections. This will allow different viewers to be used based on an object's collection membership.

2.3 Object Class Browser

The object class browser will list the object classes for which and instance of the class (i.e., object) exists in the users's search domain. the user will be able to display the object class attributes *metadata fields) for an object class,. The object class will allow the user to display and browse all the objects which exist in the user's search domain for a selected object class. The object class browser will allow the user to list and browse al the objects which exist in the user's search domain for a selected object class and sub-classes. The list of objects will be displayed in the object browser. The user will then be able to exercise all the functions of the object browser for inspecting the selected objects.

2.4 Natural Language

Natural language search allows the user to query the search domain for objects using words and sentences. The natural language window provides the user with text editor to enter and edit the query string before submitting the query to the repository. The user may also change his search strategy and thus his search domain., Upon completion of the search the user will be informed of the number of objects found which match the criteria of the search and will allow the user to either view the list of objects or ignore the results. If the user chooses to view the objects found, the objects will be displayed in the object browser. The user will then be able to exercise all the functions of the object browser for inspecting the selected objects.

2.5 Pattern Matching

Pattern match searching allows the user to query the search domain for objects by object class and metadata field value. The pattern matching window provides the user with a list of metadata fields available in the selected object class. The pattern matching window allows the user to select an object class from a list of the object classes. The list is composed of object classes for which an instance of the class (i.e., object) exists in the user's search domain. The metadata fields of the object class selected is displayed in the metadata field list. The user may then select a metadata field, enter a field value, and submit the query to the repository. Upon completion of the search, the user will be informed of the number of objects found which match the criteria of the search and will be allowed to either view the list of objects or ignore the results. If the user chooses to view the objects found, the objects will be displayed in the object browser. The user will then be able to exercise all the functions of the object browser for inspecting the selected objects.

2.6 Boolean Searching

Boolean searching allows the user to query the search domain of objects by object class and metadata field value. Boolean searching allows the user to build and submit complex

queries using “AND”, “OR”, and “NOT” operators. The boolean searching window provides the user with a list of metadata fields available in the selected object class. The boolean searching window allows the user to select an object class from a list of the object classes. The list is composed of object classes for which an instance of the class (i.e., object) exists in the user’s search domain.
The metadata fields of the object class selected is displayed in the metadata field list. The user may then select a metadata field, enter a field value, and submit the query to the repository. Upon completion of the search the user will be informed of the number of objects found which match the criteria of the search and will be allowed to either view the list of objects or ignore the results. If the user chooses to view the objects found, the objects will be displayed in the object browser. The user will then be able to exercise all the functions of the object browser for inspecting the selected objects.

2.7 Applications

The Applications menu lists all applications which are available to the user on the host system. The implementation of security for applications is based on security requirement two (2) which states:

Allow inclusion of applications which can be executed only by a controlled group of users.

The applications which appear in the Applications pulldown menu will vary based on the user’s group membership. Public users will see a set of applications which will be available to all users of the system. Secure users will see all the applications which are available to public users and those applications which have been made available to secure user based on their group membership.

3.0 Administration

The integration of security into NELS will affect many of the administrative functions. Administration of the NELS repository is performed by users known as librarians. Librarians are responsible for system administration and maintenance. Librarian responsibilities in the new system will include maintenance of applications, archival services, collections, groups, librarians, objects, object classes, and users.

3.1 Applications

The Applications Manager window will support administration activities for:

1.) Add programs available on the host system to the Applications menu.

2.) Delete programs from the Applications menu.
3.)Update the information of programs on the Applications menu.
4.) Add viewers which can be used to view objects stored on-line.
5.) Delete viewers from use.
6.) Update the information of viewers used to view objects stored on-line.
7.) Add output requests to the NELS system.
8.) Delete output requests from the NELS system.
9.) Update output requests from the NELS system.
10.) Associate an application with a group.
11.) Disassociate an application with a group.
12.) Associate a viewer with a collection.
13.) Disassociate a viewer with a collection.

The Application Manager window will continue to support all current features. The Application manager window will support two(2) additional functions which are described below.

3.2 Group List

The group list will be part of the Applications Manager window and will display the group membership list of the resident application selected. Add and delete function will be supported for the group list to allow librarians to administer group member ship for the application. T Applications Manager window will display a list of groups so that librarians may select and add one or more groups to the resident applications group membership list. The Application Manager will allow groups to be selected in the group list and deleted from the resident applications window group membership list. The second new feature is collection viewers.

3.3 Collection List

The collection list will be part of the Applications Manager window and will display the collections associated with a viewer. Add and delete functions will be supported for the collection list to allow librarians to administer collection associations for the viewer. The Application Manager window will display a list of collections so that librarians may select and add one or more collections to the viewers collection association list. The Applications Manager window will allow collections to be selected in the collection list and deleted from the viewers collection list. Collection viewers will be invisible to the end user.

3.4 Archive Manager

The Archive Manager function are composed of four (4) parts, Import, Export, Export and Delete, and Export All. the functions of the Archive Manager will require updating to include the ability to import and export group association information between groups and collections

3.5 Import

The Import function will require modification to support groups and object IDs. Information about groups which have associations with collections begin imported will need to be sorted in the import file. Import will also have to be modified to use object ID information which may be contained in the import file. The user interface to the import function will need to be updated to allow the librarian to toggle between using or ignoring the object ID information.

3.6 Export

The Export function will require modification to support saving group and object ID information. Group information need only be exported whin a collection being exported has group associations. The ID of an object should always be exported as part of the object's information.

3.7 Export and Delete

The Export and Delete function will require the same modifications as the Export function. The Export and DElete function otherwise will perform exactly as previously implemented.

3.8 Collection Manager

The Collection Manager supports the management of collections within the repository. Functions supported by the Collection Manager include "Add", "Delete, "Move" and "Modify" collections. The support group associations, a list will be added to the Collection Manager screen which will list all groups associated with a collection. "Add" and "Delete" functions will have to be added to allow librarians to administer group associations. To support collection viewers, a list will be added to the Collection Manager screen which will list al the viewers associated with a collection. "Add" and "Delete" function will be added to allow librarians to administer collection viewer associations.

3.9 Group Manager

To support the administration of groups within the NELS repository a Group Manager screen will have to be created. The Group Manager support the management of groups within the repository. Information displayed on this screen include:

Name: The name of the group.
Description: A description of the purpose of the group.
requirements: A description of the requirements which need to be met for a user to be a member of the group.
Applications: A list of applications associated with the group.
Members: A list of users who belong to the group

Functions supported by the Group Manager screen include “Add”, “Delete”, and “Update” group information. The Group Manager screen will contain a list of applications associated with the group. “Add” and “Delete” function will be supported for the list of associated applications. Any application which is associated with a group will be displayed on the “Applications” menu of the users who are members of the group. The Group Manager screen will contain a list of user who are members of the currently displayed group. “Add” and “Delete” functions will be provided to support the daministration of the group membership list. As part of that support, librarians will be able to display a list of all userws known to the system and to select one or more of the user for membership in the group. Librarians will also be able to display a list of known system groups, From this list the librarian may select one or more groups of users for membership in the current group. This feature will allow librarians to add large numbers of user to populate new groups.

3.10 Librarian Class Manager

The Librarian Class Manager allows librarians to Add, Delete, and Update librarian classes. A librarian class is a selected set of available librarian priviledges. The implementation of security in NELS will add six (6) new privileges to the Librarian Class Manager screen. These privileges include:
1.) Add group.
2.) Delete group.
3.) Update Group.
4.) Add user.
5.) Delete user.
6.) Update user.

3.11 Librarian Manager

The Librarian Manager allows librarians to Add, Delete, and Update librarians. A librarian is a user who has the ability to administer the repository based on his assigned privileges. The implementation of security in NELS will add six (6) new privileges to the Librarian

Manager screen. These privileges are:
1.) Add group.
2.) Delete group.
3.) Update Group.
4.) Add user.
5.) Delete user.
6.) Update user.

3.12 Object Class Manager

The Object Class Manager allows librarians to Add, Delete, and Update Object Classes. The Object Class Manager also allows librarians to Add, Delete and Update enumeration types. The implementation of security will not affect the user interface of the Object Class Manager. Changes to the semantic structure of the repository will make ti possible to create multiple class hierarchies. This functionality will have to be supported in the Object Class Manager, but will not affect the current user interface.

3.13 Object Manager

The Object Manager supports the management of objects within the repository. Functions supported by the Object Manager include "Add", "Delete", and "Modify" objects. The implementation of security will not change the user interface of the Object Manager.

3.14 User Manager

To support the administration of users within the NELS repository, a user manager screen will have to be created. The User Manager supports the management of users within the repository. Information displayed on this screen includes:

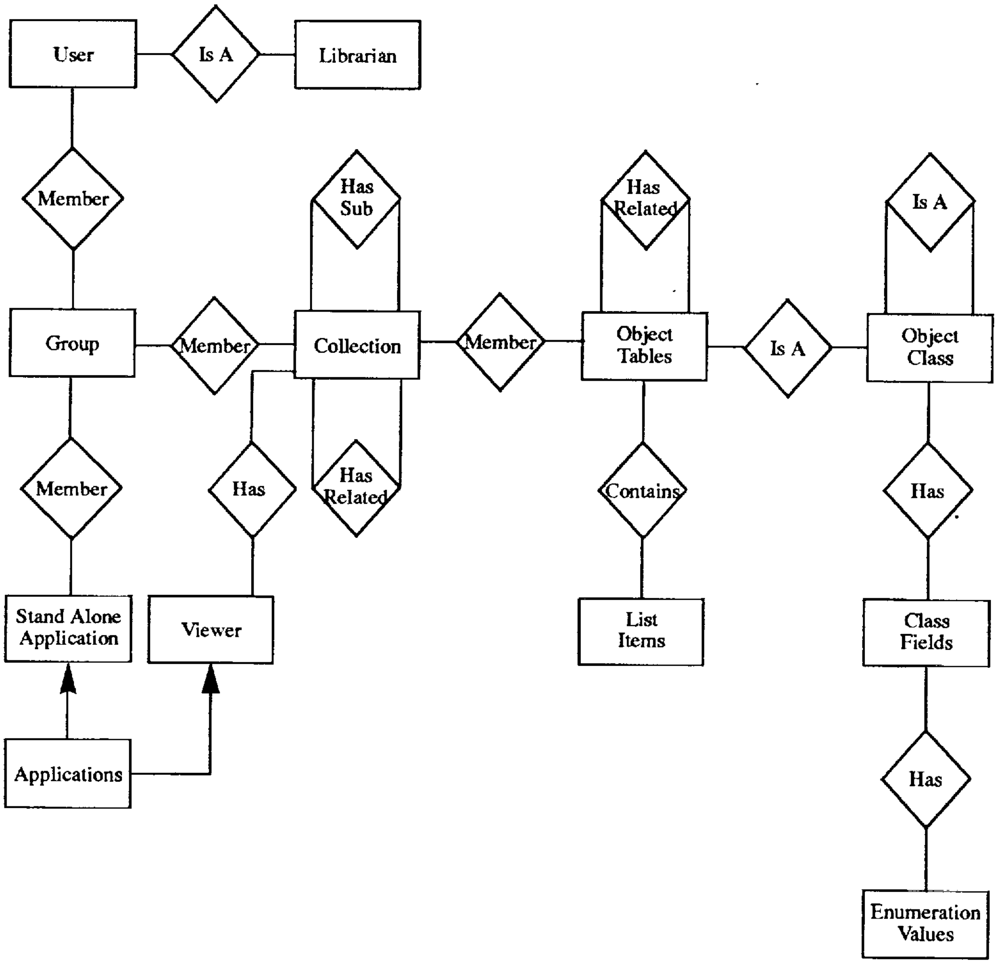
Name: The system name of the user.
Full Name: The full name of the user.
Organization: The name of the company the user represents.
Address: The mailing address of the user.
Internet Address: The internet mail address of the user.
Phone Number: The number where the user may be contacted during normal working hours.
Description: Information and notes regarding the user
Groups: A list of groups where the user is a member.

Functions supported by the User Manager screen include "Add", "Delete", and "Update" of user information. The User Manager screen will contain a list of groups where the user is a member. "Add", and "Delete" functions will be provided to support the administration of the group membership list. As part of this support librarians will be able to display a list of all groups known to the system and select and add one or more of the groups to the group membership list. Librarians also will be able to select one or more of the groups displayed in the group list and delete the selected groups from the group member ship list. Adding or updating a users's information will include administration of the users group memberships.

4.0 Database Semantic Structure

This section discusses the planned changes to the underlying structure of the NELS repository. The semantic structure of the repository is driven by the security requirements of the next release of the NELS system and is begin influenced by the desire to create a platform which could be used as a stepping stone to future releases of the system.
The diagram shows the new semantic structure of the database.

ASV4 Entity-Relationship Diagram



One of the ideas influencing the design of the database is the desire to become.database independent. the system should be structured such that it is portable to most any database system, independent of whether it is a relational or object oriented database. One limitation of the current system is the limit on the number of metadata fields in an object class. The new design will allow greater flexibility in the creation of object classes by removing the limit in the number of metadata fields in an object class. The tables and relationshi9ps for the new design are discussed below.

4.1 User

The user entity maintains information about user of the library system. The user entity has a relationship with the group entity. The relationship between the user entity and the group entity supports controlling access to collections and applications. Users may be members of zero or more groups and allow the user to have access across zero or more security groups.

4.2 Group

The group entity maintains information about groups in the library system. The group entity is the cornerstone of the security scheme. Groups work very much like they do in the UNIX operating system. The group entity has a relationship with the user entity. Users who share common privileges are put together into a group. The group entity has a relationship with the collections entity. A collection may be a member of a group. Users have access to all collections in which they share a group membership. Users who do not share this relationship will be denied visibility and access to the collection. In this scheme, security will be implemented by first creating registered users, and collections. The users and collections are then organized into groups. The group entity also has a relationship with the applications entity. A resident application may be a member of a group. Users who are a member of the same group as the resident application will have visibility and access to the application. Users who do not share this relationship will be denied access to.the application.

4.3 Application

The application entity maintains information concerning resident applications, output requests, and viewers. The application entity in the role of resident application has a relationship with the group entity. This relationship supports secure applications by allowing applications to be members of groups. The users who are members of the same group as the application will have visibility and access to the application. The application entity in the role of viewer has a relationship and collections. ‘A viewer may be associated with a collection as a replacement for the viewer normally called to display an object file. The association between a viewer and a collection is independent of the security scheme discussed thus far.

4.4 Collection

The collection entity maintains information concerning collections and the collection hierarchy. Collections have a relationship with groups. A collection may be a member of zero or more groups. This relationship between groups and collections support secure collections. If a collection is a member of a group, only member user who will have visibility and access to the collection are members of the same group.

Collections have a relationship with applications in the role of viewer. A collection may have zero or more viewers associated with it. Viewers which are associated with collections replace the default viewer for the supported object format in that collection.

Collections have a relationship with other collections. The collection hierarchy is supported by collections having parent/child relationships with each other. At the top of the hierarchy tree there is one parent collection. All other collections in the hierarchy are children of the top or root collection. A collection may be the child of one collection and the parent to may other collections. in a tree structure like this, when a user is traversing down the hierarchy tree and encounters a collection which is not accessible to them, all child collections which exist below the collection are also inaccessible. Therefore, security extends from the first secure collection down that branch of the tree to the leaf collections of the branch.

Related collection are supported by collections having relationships with other collections., These relationships exist outside4 the constraints of the hierarchical structure of the collection tree. Since these relationships are not subject to the constraints of the collection hierarchy, it is possible for a public (unsecure) collection to have secure related collections, as well as public (unsecure) related collections. To the user viewing a collection list of related collections, the list should consist only of those collection which are normally visible to the user. This policy maintains the librarians ability to establish meaningful relationships among collections and does not compromise the security constraints of the user or the repository. The identification of collections is done with a unique collection identification code. The identification code is a alpha-numeric code which starts the letter "S". Because of the method of identification, there may be only one collection hierarchy for any instance of the library repository.

4.5 Object Class

The object class entity maintains information concerning the definition of object classes and the structure of the object class hierarchy. To support the concept of a class hierarchy, the object class entity supports parent.dchild relationships among the classes. The object class entity has a relationship with the class fields entity. An object class will have one or more attributes or metadata fields,. the object class entity also has a relationship the object class instance entities, For each object class described in the object class entity there is an object class instance entity in the repository with the object class mane and containing all the fields described for that class.

Because objects classes are identified by name and on a later or number, there can be more than one class hierarchy described in the object class entity.

4.6 Class Fields

The class fields entity maintains information concerning the fields which compose the metadata attributes of an object class. Class fields may be of many different fundamental types., One of these fundamental types is an enumeration type. To support this type, the class fields entity has a relationship with the enum values entity.

4.7 Enum Values

the enum values entity maintains information concerning enumeration types. Enumeration types are user defined types composed of a set of labels. Each enumeration type defined is uniquely identified by the type name. since all information about an enumeration type is enumeration type may b used in multiple class field definitions.

4.8 Object Class Instance Entities (Object Entities)

The object class instance entities maintain information about objects stored in the repository. Each object class instance entity is an instance of an object class. The name of the object class instance entity is the same as the object class instanciated by the creation of the object class instance entity. Some of the fields defined in an object class are list fields. a list field is a field in which more than one value may exist for a single object definition. The objects stored in the object class instance entities have a relationship with collections. Objects are members of collections. One object may be a member of many collections and many objects may be members of one collection.

4.9 List Fields

The list fields entity maintains object metadata information for the list fields on an object class instance entity. Each list field in an object class instance entity will have a separate list fields entity. List field entities will be created on demand as new list fields are created in the object class hierarchy. Object classes are defined in a hierarchical manner, meaning that each new sub-class inherits all the class fields of its parent classes. If the parent class contained list fields, then an instanciation of the child class will inherit and use the list field entities of its parent. Any new list fields defined in the child class will be created when the child class is instanciated (i.e., create the object class instance entity) and will be shared by all sub-classes of the object class. The reuse of the list field entities is based on the idea that the instanciation of a class is also the instanciation of all its ancestor classes.

The reuse of list fields entities will minimize the number of tables created in the repository.

4.10 Synonym

The synonym entity maintains information linking keywords with synonyms. Synonyms are words having the same meaning as its associated keyword in most or all senses. A keyword may have one or more synonyms. The synonyms table will be used by the searching facilities of the NELS system.

4.11 Keyword

The keyword entity maintains a list of keywords used by the system. A keyword is a meaningful term used to describe an object. An object may have zero or more keyword.